MySQL - gestion du cache

Gestion du cache MySQL

C'est une méthode rapide pour lancer un second serveur pour le tester. Le plus agréable de cette méthode est que les variables d'environnement vont être adoptées par les logiciels clients que vous invoquerez avec le même Shell. Par conséquent, les connexions seront automatiquement dirigées vers le nouveau serveur. Pour les scripts de démarrage, la commande doit être :

mysqld_safe –defaults-file=path-to-option-file

Cache de requêtes MySQL

MySQL server bénéficie d'un cache de requêtes. En fait, le cache sauvegarde le texte d'une requête SELECT avec le résultat qui a été envoyé au client. Si une requête identique est appelée par la suite, le serveur retournera le résultat à partir du cache plutôt que d'analyser puis exécuter la requête à nouveau.

Le cache de requêtes est extrêmement utile dans un environnement où les tables ne changent pas souvent, et que vous avez de nombreuses requêtes identiques. C'est la situation classique des serveurs Web, qui génèrent beaucoup de pages dynamiques à partir du même contenu.

Note : Le cache de requêtes ne retourne pas de données périmées. À chaque fois que les données sont modifiées, les entrées correspondantes dans le cache sont effacées.

Si vous ne voulez pas utiliser le cache de requêtes paramétrez **query_cache_size** à zéro. En désactivant le cache de requête, il n'y a aucune surcharge apparente. (le cache de requêtes peut être désactivé à l'aide de l'option de configuration **–without-query-cache**)

Fonctionnement du cache de requêtes

Les requêtes sont comparées avant analyse, donc ces 2 requêtes sont considérées comme différentes :
SELECT * FROM tbl_name
Select * from tbl_name
Les requêtes qui utilisent différentes bases de données, différents protocoles ou différents jeux de caractères sont alors considérés comme différentes.

Si un résultat de requête a été retourné depuis le cache de requête, la variable **Com_select** ne sera pas incrémenté, mais **Qchache_hits** le sera.

Si une table change, toutes les requêtes mise en cache qui utilisaient cette table sont retirés.

Une requête ne peut être mise en cache si elle contient l'une des fonctions suivantes :

BENCHMARK()	CONNECTION_ID()	CURDATE()
CURRENT_DATE()	CURRENT_TIME()	CURRENT_TIMESTAMP()
CURTIME()	DATABASE()	ENCRYPT() avec un paramètre
FOUND_ROWS()	GET_LOCK()	LAST_INSERT_ID()
LOAD_FILE()	MASTER_POS_WAIT()	NOW()
RAND()	RELEASE_LOCK()	SYSDATE()
UNIX_TIMESTAMP() sans paramètre	USER()	

Une requête ne sera pas mise en cache dans ces conditions :

- Elle contient des fonctions définies par l'utilisateur : UDF.
- Elle contient des variables utilisateur.
- Elle fait référence à des tables de la base mysql.
- Elle est de l'une des formes suivantes :
- SELECT ... IN SHARE MODE
- SELECT ... INTO OUTFILE ...
- SELECT ... INTO DUMPFILE ...
- SELECT * FROM ... WHERE autoincrement_col IS NULL
- La dernière forme n'est pas mise en cache, car elle est utilisée comme palliatif pour ODBC, afin d'obtenir la dernière valeur insérée.
- Elle utilise une table TEMPORARY.
- Elle n'utilise pas de tables.
- L'utilisateur a un droit de niveau colonne pour l'une des tables impliquée.
- Avant la lecture de la requête dans le cache de requête, MySQL vérifie que l'utilisateur a les droits SELECT pour toutes les bases de données impliquées. Si ce n'est pas le cas, le résultat n'est pas utilisé.

Options relatives au cache de requêtes utilisés dans une requête SELECT

SQL_CACHE Le résultat de la requête est en cache si la valeur de la variable **query_cache** est à **ON** ou **DEMAND**

SQL_NO_CACHE Le résultat de la requête n'est pas mis en cache

Exemples

SELECT SQL_CACHE id, name FROM customer ;
SELECT SQL_NO_CACHE id, name FROM customer ;

Configuration du cache de requêtes

Pour configurer la taille du cache, modifier la variable query_cache_size. Une valeur de 0 désactive le cache.

L'allocation de mémoire pour le cache se fait en bloc à mesure des besoin, c'est couteux en temps, pour définir une taille de bloc modifier **query_cache_min_res_unit** (défaut : 4ko). Le nombre de blocks libres et de requêtes supprimées pour libérer de la place sont stockées dans le variables **Qcache_free_blocks** et **Qcache_lowmem_prunes**

Chaque requête a besoin au minimum de 2 blocs, un pour le texte de la requête et un autre, ou plus, pour le résultat. De même chaque table utilisée par une ou plusieurs requêtes a besoin d'un bloc.

query_cache_type modifie son comportement :

0, OFF Empêche la mise en cache ou la lecture de résultat en cache

1, ON Permet le cache, sauf SELECT SQL_NO_CACHE

2, DEMAND met tout en cache

Pour savoir si le cache est actif :
SHOW VARIABLES LIKE 'have_query_cache' ;
Pour désactiver le cache pour une session
SET SESSION query_cache_type=OFF
Pour défragmenter le cache de requêtes
FLUSH QUERY CACHE
et
FLUSH TABLE
Pour visualiser les performances du cache
SHOW STATUS LIKE 'Qcache%' ;
Le nombre total de commandes SELECT vaut
Com_select + Qcache_hits + requêtes avec une erreur
La valeur de Com_select est
qcache_inserts + qcache_not_cached + erreurs de droits d'accès ou de colonnes